ML for Healthcare Project 2

Acknowlegment

We expect everything to work on an isolated python environment created as per the instructions below, but in case you face any issues running the code please feel free to contact us by email or on MS-Teams (irodrigu@student.ethz.ch, kumarsh@student.ethz.ch, neumannam@ethz.ch).

We have tested our code in an environment with the following specifications:

- Machine:
  - CPU: `11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz`
    - `x86_64`
  - RAM: 16 GB
- OS: `Ubuntu 20.04.4 LTS`
- Python Version: `3.7.11`

Besides this, all the model training (for task 2 and 3) was done on a node with GPU (`NVIDIATITANRTX`).

Text preprocessing may take long time, so we suggest downloading the preprocessed texts from the drive link shared later in the document.

Creating isolated execution environment

- Go to the root directory (after extractig the zip)
- Execute the following in sequence (enter yes when prompted):

```
conda create -n ml4hc_proj2 python=3.7.11
conda activate ml4hc_proj2
pip install -r src/requirements.txt


# Download spacy model used for text processing (see `text_processing.py`)
python -m spacy download en_core_web_lg
```

- Now the environment should be ready
- Make sure to check that the environment is activated before running the code

Raw data/ Processed data/ Trained Models

For your convenience we are sharing the drive link containing the resources for reproducing the results.

Please visit here for the resources:
https://drive.google.com/drive/folders/1Urq0BorNnwAkshpvvoVlvvbBP-AQBu6P?usp=sharing

Following is a brief summary of the files we have made available in the drive: *(Please make sure to extract these files when needed to the path indicated later in the document in respective task sections)*

- `task_2`
  - `ml4hc_nlp_200k_raw_pubmed_data.zip` (raw pubmed dataset , 200k)
  - `ml4hc_nlp_200k_processed_data.zip` (processed pubmed texts for learning embedding and training classifiers)
  - `ml4hc_nlp_200k_embedding_model.zip` (trained Word2Vec model and genetrated dictionary and other helper files)
  - `ml4hc_nlp_200k_models.zip` (trained classifiers along with test groundtruth and prediction files and tensorboard logs)
- `task_3`
  - pretrained_BERT.zip (emilyalsentzer/Bio_ClinicalBERT pretrained model)
  - classifier_BERT.zip (Bio_ClinicalBERT with trained classification layer)
  - pooling_BERT.zip: (Bio_ClinicalBERT with finetuned output pooling layer)
  - attention_BERT.zip: (Bio_ClinicalBERT with finetuned pooling+last attention layer)

> Please refer to the file **SAMPLE_FOLDER_STRUCTURE.txt** to see the detailed folder structure.

Indications

Before running the models, please make sure to download the 200k data from the following link: [https://github.com/Franck-Dernoncourt/pubmed-rct/tree/master/PubMed_200k_RCT], or you may also use `ml4hc_nlp_200k_raw_pubmed_data.zip` shared in the google drive. Then the data should be put in a separate directory than the models, called `resources`. The data should contain 3 files: `dev.txt`, `train.txt`, and `test.txt` corresponding to the validation, the training and the test dataset respectively.

Task 1

To get the results you can run the file model_baseline directly from the terminal :

```
python src/model_baseline.py
```

The training of the model will be done and the best model should directly be running. The output will be two confusion matrices, for the validation dataset and the test dataset.

Training of the model

We already searched for the best hyperparameters which we have used in the script, so that it will take less time running the file. To tune the model and find the best hyperparameters, the lines 118 to 126 can be uncommented and the best hyperparameters should be printed on the terminal. Then you can input the new parameters to the constuctor in order to predict the results for the test data.

Results - Confusion matrix

The confusion matrix will be plotted on the screen as an output. The image reports the total number of times a label was predicted as any other class of label. For example: it could be that for the label: RESULTS, the model predicted it as RESULTS 90 times, as CONCLSUION 50 times and as BACKGROUND 20 times *etc.*

Task 2

Creating processed corpus (Similar to files in: resources/processed_data/ (Please refer: SAMPLE_FOLDER_STRCUTURE.txt))

First make sure that spacy's `en_core_web_lg` model is downloaded. (`src/text_processing.py` will try to download this automatically when running `src/corpus_generator.py`)

To create processed corpus for training the embeddings and learning classification model (for task 2), run the following:

```
python src/corpus_generator.py -o <output directory path>
```

e.g. `python src/corpus_generator.py -o resources/processed_data`

> NOTE: This would **replace** the existing files `resources/processed_data`

This will create following files:

```
<output directory path>
|
├── processed_dev.txt
├── processed_test.txt
├── processed_train.txt
├── text_original_lower.txt
└── text_processed_for_learning_embedding.txt
```

Files with `processed_` prefix , have label and processed text pairs, while others will have just processed texts.

You may use the file `ml4hc_nlp_200k_processed_data.zip` shared in the google drive to get the final processed data for next steps.

Training and testing Word2Vec Model

- **Training the embedding model**
  - The file `resources/processed_data/text_processed_for_learning_embedding.txt` created in the previous step is used for training the embedding model (by default)
  - To create trained embedding model (Word2Vec)

    ```
    python src/learn_embedding.py
    ```

This will train Word2Vec model (with vector size 200, and other default parameters)

- If you want to change the input corpus, output path, vector size, epochs *etc.*, then pass them as arguments.
- Run `python src/learn_embedding.py -h` for argument information.
- You may use the trained embedding model shared in the google drive `ml4hc_nlp_200k_embedding_model.zip` for the next steps.

- **Testing the embedding model**
  - Run:

```
python src/test_embeddings.py
```

or

```
python src/test_embeddings.py | less
```

  - It will load the embedding model from `resources/saved_models/embedding.model`, and using this model it will print out a list of similar words for a few test words like `ecg`, `doctor` *etc.*, and after that it will print out analogy results for some word triplets *e.g.* `woman->girl::man->?`

Training the classifiers

- To start training execute:
  -
```
python src/trainingutil.py --config <path-to-run-config-file>
```
  - *e.g.*

```
python src/trainingutil.py --config
src/experiment_configs/exp_02_task2_ann.yaml
```

  - The **src/experiment_configs** directory contains other configs as well, that we have used for running our experiments. You can choose any of those or create your own.

The steps above will do the following:

- It will start training
- create `runs` folder if not already present

- create a timestamped folder with `tag` value provided in the config as suffix *e.g.* : `2022-04-23_154910__exp_02_task2_ann`
  - this folder will be used to keep track of the model checkpoints, best model etc.
  - in this folder `logs` subfolder will be created in which tensorboard logs will be saved.
- the best model will be saved if the validation F1 (weighted) has increased when compared to the last best F1. Test F1 is also printed in the logs , but validation F1 is used for selecting the best model.

These are the training Config File used for different experiments (training)

Config File	Experiment description
`exp_02_task2_ann.yaml`	Fully conncted neural network
`exp_02b_task2_ann.yaml`	Fully connected neural network (with class weighting used)
`exp_03_task2_ann_unfrozen_embeddings.yaml`	Fully connected neural network (with embedding also being fine tuned)

- The models we trained are available in the shared file : `ml4hc_nlp_200k_models.zip` (in google drive)

Evaluation of saved classifier

To evaluate the models the script `src/evalutil.py` and the configs in `src/experiment_configs/eval` can be used.

e.g.

```
python src/evalutil.py --config
src/experiment_configs/eval/eval_02_task2_ann.yaml
```

This should print the scores on valdiation and test datasets.

- Make sure that the correct checkpoint path is set in the config file (under the field : `checkpoint_path`)
- The eval configs already available in `src/experiment_configs/eval` would work without any change if you use the models we shared : `ml4hc_nlp_200k_models.zip` (in google drive)

Task 3

Using pre-trained BERT

To complete this task we have used the emilyalsentzer/Bio_ClinicalBERT pre-trained BERT model available in Hugging Face.

- To train/finetune the model and obtain test results (including the confusion matrix) run:

```
python src/transformer_pipeline.py --config <path-to-run-config-file>
```

- You can also find the different configuration files used at **src/experiment_configs**

These are the training Config File used for different experiments

Config File	Experiment description
`exp_05_task3_bert.yaml`	Frozen Bio_ClinicalBERT, train classification layer
`exp_06_task3_pooling.yaml`	Finetune Bio_ClinicalBERT's output pooling layer
`exp_07_task3_attention.yaml`	Finetune Bio_ClinicalBERT's from last attention layer
`exp_mini_task3_bert.yaml`	Test execution on a small fraction of data

- After running any of the above configs for the first time, the pre-trained emilyalsentzer/Bio_ClinicalBERT model should be downloaded. If there is a network connection error, you can download the `pretrained_BERT.zip` from the drive folder, unzip it and run:

```
python src/transformer_pipeline.py --config <path-to-run-config-file> --pretrained <path-to-pretrained-model>
```

Evaluation of saved classifier

In the previous drive you can also find the language models we finetuned. To evaluate them and obtain accuracy and f1 scores you can uncompress the zip files, copy the contained folder into `resources/saved_models`, and run `src/transformer_pipeline.py` with the respective config in `src/experiment_configs/eval`. *e.g.* for evaluating classifier_BERT.zip

```
python src/transformer_pipeline.py --config
src/experiment_configs/eval/eval_05_task3_bert.yaml
```
